Why care about computers?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Why it's useful to learn this

- Although everyone uses computers, few people actually know how they work
- Programmers are very well paid
- If you want to learn how to program, you need to know the basics of computers first
- It will help you understand how computers communicate

See you next lecture!

What is a computer?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What computers are and what they do

- A computer is anything that can compute, which means it can determine an amount or number when given certain rules
- Things that are technically computers: calculators, watches, wearables, cash registers, etc.
- Your brain is also a kind of computer

- Computers can solve problems
- They take instructions, run calculations, and give you a solution
- Computers are all around you

See you next lecture!

How does a computer work?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How computers do what they do

- Without humans to operate the computers, they're just dumb machines that don't do anything
- By giving them specific instructions and commands, humans are communicating with the Central Processing Unit (CPU), the core of the computer, where the problem gets solved
- CPUs = tiny silicon boxes attached to the motherboard
- There are only two companies that make CPUs for personal computers: Intel and AMD
- Within these squares there are thousands of transistors that are microns in diameter
- A transistor is a wire that can transfer electricity
- The smallest CPU transistor is 0.05 microns

- A processor is like a small calculator. It transfers electricity in circles, which we call a clock, and is measured in MIPS (Millions of Instructions Per Second)
- Computers have permanent and temporary memory
- Permanent memory is build in the form of a hard drive which stores your files
- Temporary memory is called RAM (Random Access Memory), and here the information is stored just for the duration of the program you're running

See you next lecture!

How do computers send & receive information?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How computers communicate

- In the world of computing there are tons of peripherals: monitors, keyboards, mice, track pads, etc.
- Peripherals are external devices that allow you to give your computer input
- The computer receives the input and responds with an output on the monitor
- As a programmer, the code you write is the input that the computer receives
- The response will depend on your coding abilities

See you next lecture!

Machinespeak & The Matrix

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The language of computers

- Computers have their own language called machine code, which is really hard to understand for humans
- Peripherals and other devices communicate through machine code
- The simplest version of machine code consists of 0s and 1s (beeps and boops), also called a binary system

  1 = Yes, 0 = No

- The motherboard transfers messages to the processor by sending electricity through the wires in the board
- Letting electricity go is a 1 and stopping it is a 0
- In a second, it can send close to a million 0s and 1s
- The original computers were giant tubes, and they were using air instead of electricity (not that efficient)

- **There is a spectrum of programming code, from low level to high level code**

- Low level code:
  - used by basic system and basic pieces of hardware
  - long, inefficient, hard to read or write
  - infinitely customizable
  - machinespeak is the ultimate low level code

- Mid level code:
  - used by peripherals
  - slightly higher level than 0s and 1s
  - peripherals might require a driver to be installed
  - a driver is an interpreter that translates your input for your system

- High level code:
  - used by programmers
  - designed to make things faster
  - shorter, easier to read
  - less descriptive and more restricted

See you next lecture!

What is an operating system?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What operating systems are and the first operating system ever

- **Operating systems are interpreters that allow you to communicate with the computer**
- Before operating systems, only highly technical people could use computers
- Operating systems were the world's first software
- A software is an application or a system of instructions and logic that exists within the hardware and is used to control the hardware
- The first operating system was MS-DOS (Microsoft Digital Operating System):
  - it was a black screen where you typed in commands to do certain tasks
  - you had to remember the commands and where the programs you want to use are stored
- MS-DOS was taking your high level instructions and translating them to low level code

See you next lecture!

Operating systems in the modern era

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How operating systems evolved

- In the 1980s, Bill Gates and Steve Jobs were fighting over how to make MS-DOS better
- Steve Jobs copied the idea from Xerox, which came up with the GUI (Graphic User Interface)
- Before GUI, operating systems were text based user interfaces
- After seeing the GUI, Steve Jobs created Macintosh and packaged it in an attractive way for the average consumer

- Computers used to be popular only among hobbyists, but after operating systems came out, everybody could easily use them

- There are 3 main operating systems today: Windows, MacOS, and Linux

- Windows' mission was to make an operating system that appeals to everyone

- MacOS is even easier to use and more restricted

- Linux is free (open source), anyone can modify it, and it's used mostly by technical people who want to have more control over the software

- Other operating systems that are similar to Windows: Ubuntu and Fedora

See you next lecture!

Desktop software

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What desktop software is and what it does

- "Desktop" refers to your operating system. Your computer's desktop is where you keep files and folders
- **Desktop software = Software that you can open from your desktop**
- Software translates everything you do for the hardware
- MacOS and Windows have different machine code, so you have to make specific software for each of them
- Within the hardware of your computer runs an operating system, and the operating system runs desktop software
- All these systems have to communicate between them
- You can think of these like layers

- Apple used to make all the apps that run on a Mac, but now they give you the option to use other versions of the same tools

  They have transformed into a platform

- Platform = Software environment where anyone can create something that runs on it

- Nowadays, almost every device is a platform that you can install apps on (phone, watch, TV)

See you next lecture!

There are lots of programming blogs, Slack groups, Facebook groups, etc. --> start getting involved with the community.
Go on Hacker News and find a thread that you find interesting.
Do the same with Stack Overflow ^^^
Check out GitHub and browse through some of their power //
Go on Dribbble and search terms like "front end" and "back end"

The history of the internet

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Why internet was created and how it evolved

- The first version of the internet was made by DARPA (Defense Advanced Research Projects Agency)
- Created in the 1960s, ARPANET was a project that aimed to connect two computers to each other, one in UCLA and the other one in Stanford
- Through this connection, the world's first computer message ("LO") was sent by mistake by a clueless student who was trying to type "LOGIN"
- Later on, they created the TCP/IP (Transmission Control Protocol/Internet Protocol), a system that computers use to transfer information to and from servers and other computers
- Until the 1980s, the internet was only used for transferring research between research institutes

- LAN = Local Area Network (local computers connected to each other)
- The internet consists of many LANs connected to each other
- WWW = World Wide Web (not the same thing as the internet)
- WWW is a standardized system that was created so that any kind of computer can access and transfer the same information
- One of the ways it does that is through the HTTP protocol, which acts as a language for coding information for any computer
- When you type http://www.website.com, "http" specifies the protocol, and "www" specifies the location of what you're looking for

See you next lecture!

The anatomy of the internet

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How the internet works

- There are 3 concepts you need to know: client, node, server
- The client is the device that requests and receives information
- A node is any machine that that information crosses through to get to the client
- The server is the device that's sending the information, fulfilling the client's request
- These terms are relative and they can be interchangeable depending on which device sends or receives the information
- The most important nodes you need to understand are: ISP (Internet Service Provider), modem, router
- Any information will have to pass through these nodes to get to your computer

- The ISP is any mega hub that distributes and connects computers across the country

  Instead of connecting every computer in the area with each other, you connect all of them to this hub

- The modem is a device that is permanently connected to your ISP and is used to send and receive information through your ISP hub

- The router is a hub that allows all devices and systems in one area or room to connect to the same connection through the modem

See you next lecture!

Domain, IP, DNS

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the three concepts

- IP = Internet Protocol
- The IP is a 9 digit string of numbers appended together with periods
- Computers use IP addresses to find the location of your computer or find the server you're requesting
- Every device has an IP
- DNS = Domain Name System
- The DNS holds the information of what IP address is associated with what domain name
- ICANN = The International Corporation of Associated Names & Numbers
- ICANN is the entity that decides if the domain name you want is available

See you next lecture!

How do browsers work?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What browsers are and their functions

- A browser is a software that allows you to access websites
- It has two basic functions:

  #1 It establishes a persistent connection with the server where you can access files to view a website

  #2 It translates web programming into something that your operating system can understand
- There are several hundred browsers out there
- The most commonly used are:
  - Google Chrome
  - Mozilla Firefox
  - Internet Explorer
  - Safari
  - Opera

- Every browser has a different way of interpreting web code

  The same site might look different depending on which browser you're using to view it

- Programmers have to ensure that the website they're building will look good and load correctly on all of these browsers

See you next lecture!

How does mobile internet work?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How you get internet on your phone

- Smartphones are just small computers that ring
- The mobile phone is a client that receives information
- There are two ways of getting data on your phone: Wi-Fi and cell service
- Wi-Fi only works within a certain range
- Cell service works over longer distances and is provided by the cell towers
- The phone companies that own these cell towers act like the ISP, the cell tower acts like the router, and the modem in this case is called a Gateway Server
- The Gateway server converts data into a language your phone can understand, called WAP (Wireless Application Protocol)

See you next lecture!

The anatomy of a website

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What a website is made of

- A website is just a big set of text files in a folder stored on the server
- The folder usually contains these 3 files:
  - index.html (main instructions about what the website should show you)
  - a .css file (instructions on the style of the website)
  - a backend script (instructions of what the site can do) - .php, .py, or .rb
- Other potential files might include a folder called "images", where all the images that should be displayed on the website are stored
- For each page of the website there will be a separate folder with the same structure

See you next lecture!

The anatomy of a mobile site

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What a mobile site and a mobile app are made of

- Nowadays, almost all websites have in their original index file instructions on how to display them on mobile phones
- When accessed through the browser, mobile sites operate very much like the browser on your computer
- Mobile sites are different from mobile apps
- While you can access a mobile site by opening your browser, in order to use a mobile app you will have to download it to your phone
- The structure of the folder that holds the mobile app files is similar to the mobile site folder, only it's now on your phone instead of online

- The group of files you download is called a package

- This package doesn't have a backend file, which is stored on the server of the company that provides the app, in order for it to run faster

See you next lecture!

Let's talk about Netscape

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What Netscape was and how it change the world

- Netscape was the first commercially available browser
- It did 3 things that changed the internet forever:
  - #1 They invented a new language called Livescript that allowed websites to have richer functionality (they were very basic at first)

    Livescript later became Javascript
  - #2 They invented SSL (Secure Sockets Layer) which encrypted the traffic being passed through the TCP/IP protocol

    People could buy things online without being afraid someone will steal their information
  - #3 It launched what we call "browser wars"

- Companies started fighting about who makes the best browser

- As a result, Microsoft launched Internet Explorer

See you next lecture!

What is front-end vs. back-end?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the two concepts

- In any web interaction you have, you're going to use a browser to access information on a server
- **Front-end programming is a set of instructions that tells your browser what to show (images, text, spacing, buttons, etc.)**
- **Back-end programming is a set of instructions that does things like fetching information, saving information, and running calculations**
- The front-end instructions get processed on your browser, which uses your computer's RAM and processor

  Too many websites open will slow your computer down
- Back-end instructions run primarily on the server you accessed the information from and uses its hardware

- Back-end programming = Server-side programming
- In order for a website to run effectively, both front-end and back-end programming must operate together

See you next lecture!

FRONT_OR_BACK?

WHICH TASK IS WHICH?

The lines are often blurred when it comes to determining frontend vs. backend responsibilities because either some people can do it all or some companies don't know what they're hiring for. Take a look at the following list and see if you can distinguish who does what.

Motivated to combine the art of design with the art of programming

Develop new user-facing features

Outputting data in different formats

Build reusable code and libraries for future use

Proficient understanding of client-side scripting and JavaScript frameworks, including jQuery

Responsible for managing the interchange of data between the server and the users.

Development of all server-side logic and maintenance of the central database.

FRONT_OR_BACK?

WHICH TASK IS WHICH?

Assure that all user input is validated before submitting to back-end

Responsible for integrating the ___ end elements built by your coworkers into the application

Building reusable code and libraries for future use

Work with the UI/UX designer and bridge the gap between graphical design and technical implementation, taking an active role on both sides and defining how the application looks as well as how it works

Optimization of the application for maximum speed and scalability

Implementation of security and data protection

Good understanding of the following: advanced JavaScript libraries and frameworks (AngularJS, ReactJS, etc.)

Design and implementation of data storage solutions

Proficient understanding of code versioning tools like Git

FRONT_OR_BACK?

WHICH TASK IS WHICH?

Understanding accessibility and security compliance if necessary

Optimize application for maximum speed and scalability

User authentication and authorization between multiple systems, servers, and environments

Integration of multiple data sources and databases into one system

Good understanding of asynchronous request handling, partial page updates, and AJAX

Implementing automated testing platforms and unit tests

Ensure the technical feasibility of UI/UX designs

Proficient understanding of code versioning tools, such as Git

Management of hosting environment, including database administration and scaling an application to support load changes

FRONT_OR_BACK?

WHICH TASK IS WHICH?

Data migration, transformation, and scripting

Setup and administration of backups

Proficient understanding of web markup, including HTML5, CSS3

Understanding differences between multiple delivery platforms such as mobile vs desktop, and optimizing output to match the specific platform

Creating database schemas that represent and support business processes

Responsibilities will include translation of the UI/UX design wireframes to actual code that will produce visual elements of the application

Proficient knowledge of the following: PHP, Python, Ruby, Java, .NET, JavaScript etc.

Proficient understanding of cross-browser compatibility issues and ways to work around them

Basic understanding of JavaScript, HTML5, and CSS3

ANSWERS

Front = Back =

Motivated to combine the art of design with the art of programming

Develop new user-facing features

Outputting data in different formats

Build reusable code and libraries for future use

Proficient understanding of client-side scripting and JavaScript frameworks, including jQuery

Responsible for managing the interchange of data between the server and the users.

Development of all server-side logic and maintenance of the central database.

Assure that all user input is validated before submitting to back-end

Responsible for integrating the ___ end elements built by your coworkers into the application

Building reusable code and libraries for future use

Work with the UI/UX designer and bridge the gap between graphical design and technical implementation, taking an active role on both sides and defining how the application looks as well as how it works

Optimization of the application for maximum speed and scalability

Implementation of security and data protection

Good understanding of the following: advanced JavaScript libraries and frameworks (AngularJS, ReactJS, etc.)

Design and implementation of data storage solutions

Proficient understanding of code versioning tools like Git

Understanding accessibility and security compliance if necessary

Optimize application for maximum speed and scalability

User authentication and authorization between multiple systems, servers, and environments

Integration of multiple data sources and databases into one system

Good understanding of asynchronous request handling, partial page updates, and AJAX

Implementing automated testing platforms and unit tests

Ensure the technical feasibility of UI/UX designs

Proficient understanding of code versioning tools, such as Git

Management of hosting environment, including database administration and scaling an application to support load changes

Data migration, transformation, and scripting

Setup and administration of backups

Proficient understanding of web markup, including HTML5, CSS3

Understanding differences between multiple delivery platforms such as mobile vs desktop, and optimizing output to match the specific platform

Creating database schemas that represent and support business processes

Responsibilities will include translation of the UI/UX design wireframes to actual code that will produce visual elements of the application

Proficient knowledge of the following: PHP, Python, Ruby, Java, .NET, JavaScript etc.

Proficient understanding of cross-browser compatibility issues and ways to work around them

Basic understanding of JavaScript, HTML5, and CSS3

What is a language?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining programming languages

- Programming is a way to write instructions for a computer to understand and work with
- Computers speak machine code
- **A programming language is a set of rules you have to follow in order for your computer to understand you**
- Programming languages are similar to human languages. They are only good for software or hardware that is designed to understand them
- Programming has developed over the years as more and more people contributed to it
- Even though newer languages are more efficient, a lot of products (ex. operating systems) we use are still based on old languages

See you next lecture!

HTML

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What HTML is and what it's used for

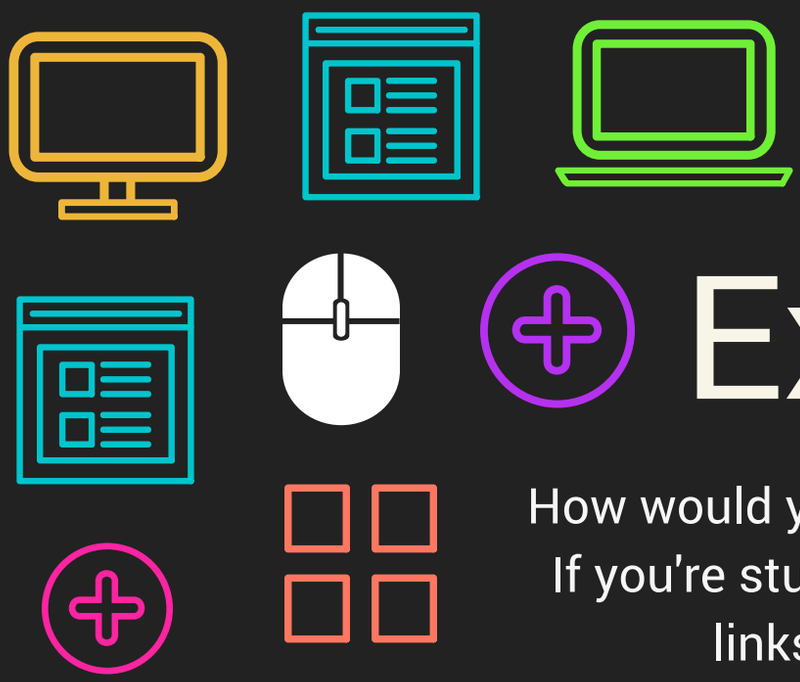
- **HTML = HyperText Markup Language**
- HTML was created by people who wanted to increase the efficiency of sharing research
- They first created the HyperText Transfer Protocol (HTTP), and with this, people could use HyperLinks in a document to embed links to other documents
- Then, the researchers wanted to markup the texts with highlights, underlining, bolding, and so HTML appeared

- HTML is by far the simplest web language you can learn and it's included in every website

- HTML is used for formatting text, tables, images, buttons, and it assigns attributes to these objects

- There are 142 tags you can use in HTML
- HTML5, the latest version, lets you embed anything you want into a page, like music, videos, games

See you next lecture!



HTML Exercises

How would you modify the HTML below? If you're stuck you can use the resource links for hints and/or check your answers.

Change the size of the image to 350 pixels wide and 500 pixels tall

```
<!DOCTYPE html>
<html>
<body>

<img src="thisisfun.jpg" width="104" height="142">

</body>
</html>
```

Change the text color of the paragraph to "black"

```
<!DOCTYPE html>
<html>
<body>

<p style="color:red">This is a paragraph.</p>

</body>
</html>
```

Add italics to the word "awesome"

```
<!DOCTYPE html>
<html>
<body>

<h1>What Does Udemy Do?</h1>

<p>Udemy's mission is to offer awesome courses.</p>

</body>
</html>
```

Transform the text below into a link that goes to https://www.youtube.com/watch?v=-w-58hQ9dLk.com/

```
<!DOCTYPE html>
<html>
<body>

Check this out.

</body>
</html>
```

Make "Tokyo" into a heading

```
<!DOCTYPE html>
<html>
<body>

<p>Tokyo is Japan's capital and mixes the ultramodern with the
traditional from neon-lit skyscrapers, to historic temples, and the
opulent Meiji Shinto Shrine.
</p>

</body>
</html>
```

CSS

LECTURE SUMMARY

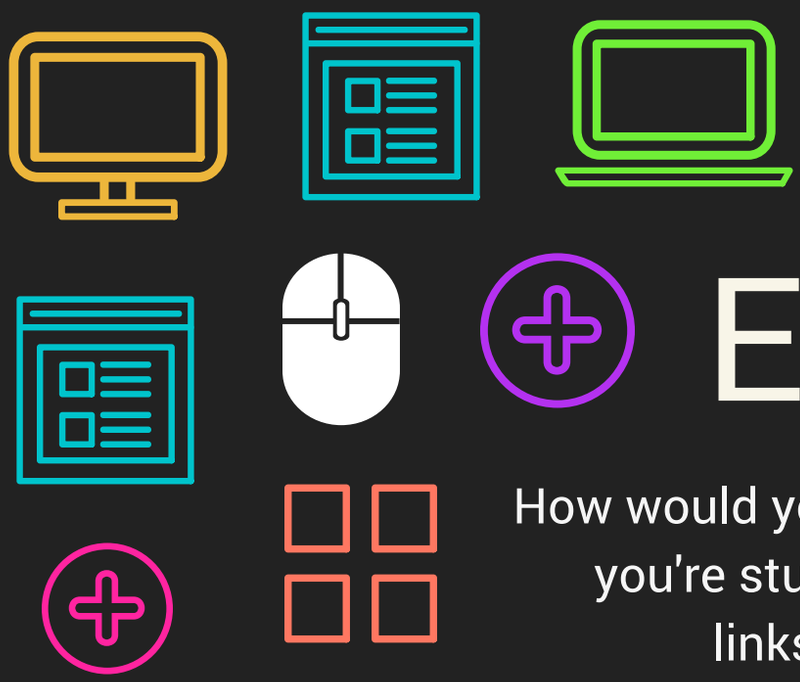
PRE-PROGRAMMING

Covered in this lecture:

What CSS is and what it's used for

- **CSS = Cascading Style Sheets**
- CSS allows you to have more control over the page than HTML does
- With CSS, you can create a variety of new attributes and apply them to HTML elements on the page, by using what's called a "class"
- For example, you can arrange elements on the page wherever you want them by describing the location
- Any attribute you give to a class will be given to whatever you wrap in that class tag
- Class rules can be created in the same document as your text, separated at the top, or in a separate .css file that will be referenced in the original document
- Every developer has to have a basic understanding of CSS

See you next lecture!



CSS Exercises

How would you modify the CSS below? If you're stuck you can use the resource links for hints and/or check your answers.

Set the background color to pink.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
background-color: lightblue;
}
</style>
</head>
<body>

<h1>Pink is my favorite color.</h1>

<p>This page now has a pink background color!</p>

</body>
</html>
```

Double the top padding(s).

```
!DOCTYPE html>
<html>
<head>
<style>
p.padding {
padding-top: 5cm;
}

p.padding2 {
padding-top: 50%;
}
</style>
</head>
<body>

<p>This is a text with no top padding. This is a text with no top padding. This is a text with no top padding.</p>
<p class="padding">This text has a top padding of 2 cm. This text has a top padding of 2 cm. This text has a top padding of 2 cm.
</p>
<p class="padding2">This text has a top padding of 50%. This text has a top padding of 50%. This text has a top padding of 50%.</p>

</body>
</html>
```

Change the CSS to float right.

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
float: left;
}
</style>
</head>
<body>

<p>In the paragraph below, we have added an image with style
<b>float:left</b>. The result is that the image will float to the
right in the paragraph.</p>
<p><img src="w3css.gif" width="100" height="140">
This is some text. This is some text. This is some text.
This is some text. This is some text. This is some text.
</p>

</body>
</html>
```

JavaScript

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What JavaScript is and what it's used for

- JavaScript is not the same thing as Java, which is a back-end language
- JavaScript was created by Marc Andreesen and it was originally called Mocha
- It's the hardest language to learn
- **JavaScript is in charge with the website interactivity**
- It was originally created for facilitating the process of filling out forms; before JavaScript, you couldn't know if the username you're trying to use was already taken
- JavaScript allows websites to run faster, it makes the site experience more interesting and enjoyable

See you next lecture!

Javascript Exercises

So now you have the lowdown on the three main languages you have to know to do any type of programming.

1. HTML: defines the content of web pages

2. CSS: specifies the layout of web pages

3. JavaScript: programs the behavior of web pages

Edit the code below so the "Click Me!" button changes the text to “After!” when you click it. Run it through the “Try Me” resource link and see what happens.

There are lots of other fun JS exercises in the "These are fun!" resource link. Go ahead, Script your heart out.

```
<!DOCTYPE html>
<html>
<body>

<h1>What Can JavaScript Do?</h1>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button"
onclick="document.getElementById(´demo´).innerHTML =
´Hello JavaScript!´">Click Me!</button>

</body>
</html>
```

Python

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The pros and cons of Python

- Python is a versatile programming language and it's easy to learn
- It's considered a general purpose programming language
- Python is an expressive language because it resembles very closely the English language
- Other languages make you learn various signs that if you forget to include, the code fails
- Python looks like this:

  if a is not 5:
- Who uses Python: Google, YouTube, Dropbox, NASA
- Python is a high level programming language, which means it's the farthest away from machine code, so it's less precise

- This makes it harder to have full control over what you're trying to do

- The downside to being a general purpose language is that every time you use it, you have to install other technologies to help interpret the code for you

See you next lecture!

PHP

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What you need to know about PHP

- PHP = PHP Hypertext PreProcessor
- PHP is one of the most well-known back-end languages and it has the largest community of developers
- It's fairly easy to learn but it has a few inefficiencies because it's so old
- PHP looks like this:
  Display "if (a!=5) {..."
- PHP is open source, anyone can use it and set it up very quickly
- PHP is the easiest to find and recruit developers for, because of its large community
- It has a negative reputation because it's free

- Although it has some drawbacks, PHP is flexible, well supported, and easy to use

- Who uses PHP: Facebook, WordPress

See you next lecture!

Ruby

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What you need to know about Ruby

- Ruby is the newest programming language, it's popular and a bit controversial
- It was designed to increase speed
- It includes a lot of automation and intuitive changes that save time
- Ruby has a small community and therefore the prices are very high for a Ruby developer
- It has a lot of inexperienced people because it's so new
- It has poor performance when used for larger and larger systems
- Who uses Ruby: AirBNB, Shopify, Etsy, Groupon

See you next lecture!

What's a tech stack?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The definition of tech stacks and examples

- The operating system acts as a giant interpreter for your computer, and as a result any software or program works within the space that it's created
- We usually think of this like a "stack"
- Any set of technology of programming languages that works together and enables each other will and can be referred to as a stack
- Web stack = The combination of technologies a website uses
- Tech stack = Web & mobile apps
- For example, a website can be built using the LAMP stack:

  Linux (operating system), Apache (server system), MySQL (database), PHP (back-end language)

- Facebook uses LAMP

- If you change one of the pieces, you can get WAMP (Windows), or MAMP (MacOS)

See you next lecture!

Common stacks for web

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

More examples of stacks

- Any web stack has 4 components: operating system, server system, database, back-end language
- LAMP is the oldest and most popular stack
- When you have different needs for your technology, you can use other stacks
- WINS = Windows Server, IIS Server, .NET, Microsoft SQL Server
- WINS is great for big enterprises, because it focuses on security and IP protection
- It's also very slow to use, expensive, and you have to train people on how to use it
- MEAN = MongoDB, Express.js, Angular.js, Node.js
- MEAN is great for small startups that need cutting edge technology, speed, and a polished user interface

See you next lecture!

Syntax

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of syntax

- All programming languages have the same core concepts, they are just expressed / written in a different way
- Every language has its own rules and its own vocabulary
- One of the first things you do when you learn programming is learn these rules
- Syntax can be strict or expressive
- Older languages have strict syntax, which means if you forget to put one sign or you capitalize the wrong letter, the code will fail
- Python, for example, will still work even if you get a few things wrong

See you next lecture!

Variables

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of a variable

- Variables are stored information that you can then manipulate and use
- They are used in order to make the sites dynamic
- If you want to change something on the website that shows up on multiple pages, you just have to modify the variable and the change will happen in all the places

See you next lecture!

Printing

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of printing

- Printing is a command that you can put into your code in order to tell it to output the result of the code you're running
- If you want the website to make a calculation, it's not enough to write the code for it, you have to "print" it on the screen if you want the user to see the actual result
- There is a specific syntax for printing in each programming language

See you next lecture!

Commenting

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of commenting

- When developers are working on an application that someone else built, they need to know some more details about the code in order to be able to modify it
- Comments are a way of adding notes to explain what you're doing
- In order for the computer to know that what you're writing is not code, you have to "comment it out" by using specific symbols for each language
- This way, it will ignore that part and the program will run properly

See you next lecture!

Strings

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of strings

- If you want to save a variable as text, you have to store it as a string
- There are different rules for each language, but usually you would use quotations to delimitate text
- The computer will consider whatever is between the quotations as human language and won't run it as code
- Escape commands allow you to modify the text within your quotations, like put in on two separate lines

  Example: "This is /n a text"

See you next lecture!

Arrays

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Explaining the concept of arrays

- Array = Series of pieces of information that are all formatted the same
- Arrays can include numbers, strings, or variables (but not combined)
- Using arrays saves time and it allows you to show more information more easily
- If you need to use only one item from the array, you use a pointer
- Pointers show what position each item of the array is in

See you next lecture!

What is a framework?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What frameworks are good for

- **A programming framework is a predefined structure that programmers can use to build upon, by changing existing code or adding code of their own to create a product**
- Most of the time, what you're trying to create has already been done before
- There is no point starting from scratch every time you build something
- Frameworks help speed up the process of creating your product
- They help you set up the basic folder structure and link everything together for you so that you can spend time working on the parts that are unique
- It's like using a template and changing only the parts you need to be different

See you next lecture!

Front-end frameworks

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The use of front-end frameworks

- Front-end frameworks are template websites that are set up in the most convenient way possible
- It's easy to modify these templates and it saves time
- Front-end frameworks might provide UI packs (UI = User Interface)
- UI packs include pre-made button designs, forms, fonts, headers, everything you see on the site
- Front-end frameworks usually provide 3 things:
  - Layouts
  - UI kits (graphic design bits)
  - Interaction snippets

See you next lecture!

Back-end frameworks

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The use of back-end frameworks

- Back-end frameworks do the following things:

  #1 They help you set up the file and folder structure on the server
  #2 They give you pre-made code for functions that you're going to use
  #3 They help you download filed
  #4 They help you optimize images, save, and recall them
  #5 They help you set up APIs for pulling information from other websites
  #6 They help you set up a payment terminal

- Back-end frameworks force you to program your app in a specific way that fits with their structure

- It can be complicated to learn full frameworks, that's why you can also use micro-frameworks

- Micro-frameworks focus on core functions

See you next lecture!

What's an IDE?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What IDEs are and what they do

- **IDE = Integrated Development Environment**
- IDEs are programs that programmers use to make their applications inside of
- They provide tools that make coding a lot easier, like spellcheck or auto-complete features
- IDEs help the developer organize and structure the files before they upload them to a web server
- IDEs also help with version control, keeping track of code updates
- Inside an IDE, you can write code, test it, and automatically see how it runs for real as if it were uploaded to a live website

See you next lecture!

Libraries

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What a library is and what it's used for

- Libraries are packages of pre-made commands that help you save time
- JQuery is one of the most popular and famous libraries for JavaScript
- Other JS libraries: Handlebars, Mustache
- There are tons of libraries out there for everything you might need
- There are libraries for front-end, as well as back-end languages

See you next lecture!

What is an API?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What APIs are and what they do

- **API = Application Programming Interface**
- APIs are mini applications that run on top of a larger application, and are used to facilitate the transfer of information to external apps that ask for them
- APIs can automatically handle the transfer and fulfil the information requests
- They help save time and they can limit what other apps can ask for
- Example of using an API: Login to a website using Facebook

See you next lecture!

CREATE WEBSITES THAT TALK TO OTHER AND COMBINE THE INFORMATION THEY RECEIVE TO CREATE A MASHUP.

API's are basically how pieces of software (websites, operating systems, etc.) interact with each other. They allow (or don't allow) for information exchanges and cooperation. Data sharing is caring.

A lot people like to use this example: At a restaurant you have a menu, it represents all the different foods that you can order. You place an order with the waiter and he returns with your choice.

Menu = API
Placing an Order = Executing an API Call
Food = the System's Response

- A website that shows trending keywords by city.
- A timeline of the world's top YouTube videos.
- A Google Map of the most Instagrammed locations.

Go to PogrammableWeb's API directory (see resource link) and try to conjure up this year's hottest API mashup. If you're stuck, see the case study link. Post your idea in the discussion board.

What is a CMS?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What they are and why we use them

- **CMS = Content Management System**
- A CMS is an application that helps you as a website owner or website developer manage your content
- Content is a very broad term, and it includes everything you create and put online (blogs, posts, images, graphs, PDFs, guides, videos)
- Without CMS, you would have to build everything from scratch in order to put your content online
- Using a CMS makes this whole process easier and it makes the site look better
- CMS handles your content, it lets you manage commenting, share information, quickly change how the site works, etc.
- You don't have to be a programmer to use a CMS

See you next lecture!

The big three CMS

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

WordPress, Drupal, Magento

- **#1 WordPress**

  - The world's largest and most popular CMS
  - 22% of the entire internet is built on it
  - It has tons of templates, plugins, and extensions
  - It can handle mostly everything (payments, email lists, post scheduling, etc.)
  - It's pretty hard to use

- **#2 Drupal**

  - It has modules, plugins, and extensions for everything you might want to build
  - Very similar to WordPress
  - It's less popular or mainstream
  - Only 2.2% of the internet is built on it
  - It has a more enthusiastic but smaller core of developers supporting it

- WordPress and Drupal are considered multi-purpose CMS

- **#3 Magento**

  - It's a CMS that handles e-commerce / online stores

  - It helps with inventory management, SKUs, and product imagery

  - It's huge and popular

- All these CMS are built in PHP, they're easy to set up, and they're mostly free to use

See you next lecture!

The new guard CMS

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

SquareSpace, Weebly, Shopify

- Even if the big three CMS are convenient to use, they still need some extra time and effort to set them up
- This is why WYSIWYG (What you see is what you get) platforms emerged

▶ **#1 SquareSpace**

- It lets you create a site within a given set of templates
- You are limited in what you can edit

▶ **#2 Weebly**

- It lets you have more customization control than SquareSpace
- You can install integrations, third party themes, and move things around
- They are less popular

- **#3 Shopify**

  - It handles e-commerce

  - You can set up a store and add all sorts of cool advanced features and integrations

  - Easier to use than Magento

  - They take a percentage of your store's sales, as well as a weekly subscription

  - Built in Ruby

See you next lecture!

Picking one over the other

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How to choose your CMS

- There are some questions you need to ask yourself before choosing the right CMS

- **#1 Are you going to sell things online?**

  - If yes, use Magento or Shopify

  - If not, use the other ones

- **#2 Is selling things online the primary purpose of your site?**

  - If you also want to have a blog, Magento will make it easier for you

  - You can use two separate CMS, one for selling, and another one for blogging

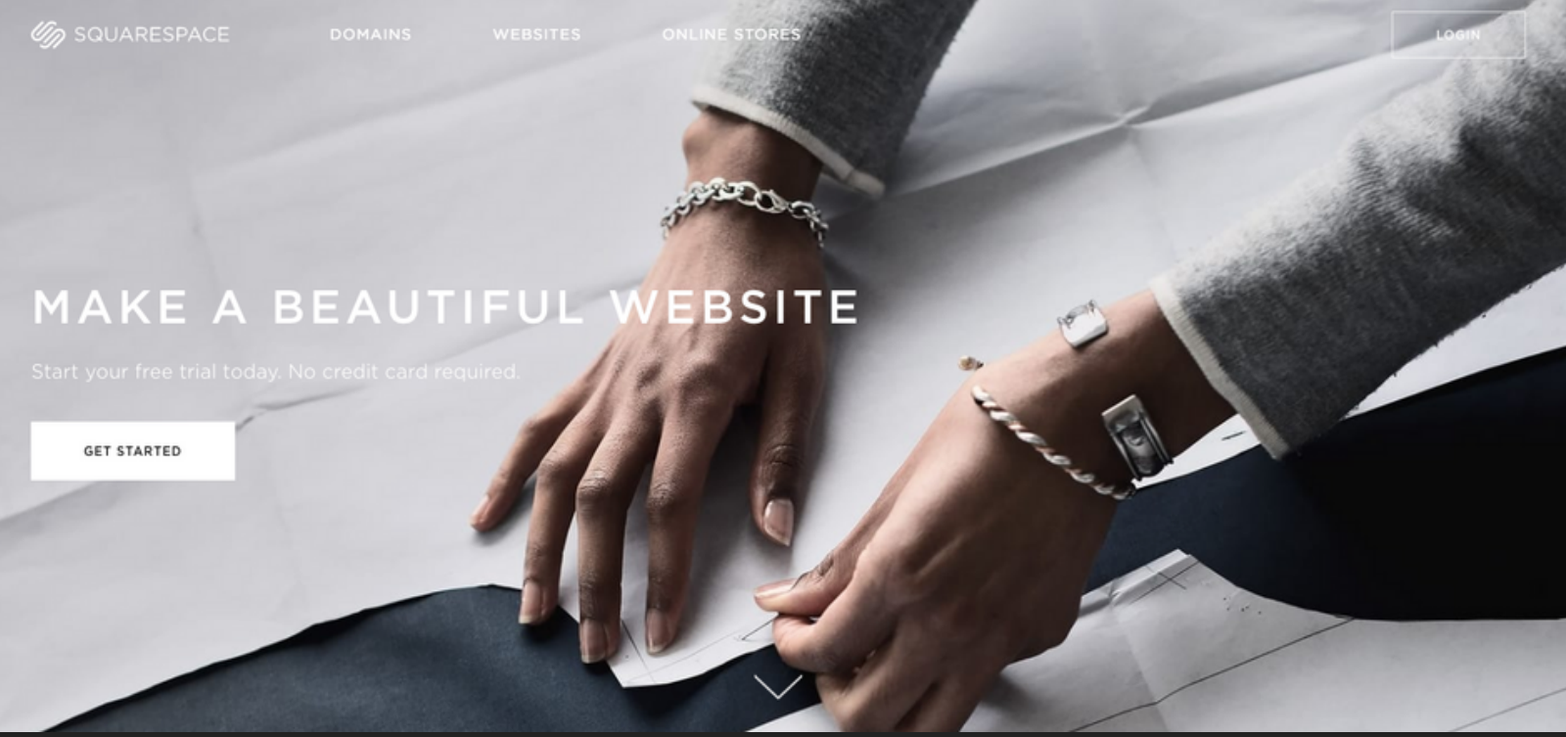
  - If selling is only one thing you do among a lot of other stuff, choose one of the multi-purpose CMS

- **#3 How particular are you about details?**

  - If you want your site to be unique, use a traditional CMS because you can customize more

  - If you're satisfied with a template design, use the new ones

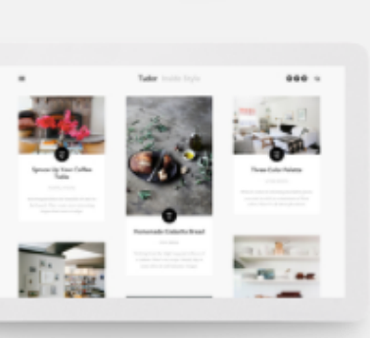
See you next lecture!

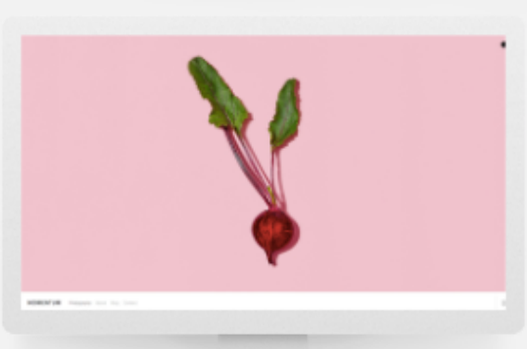


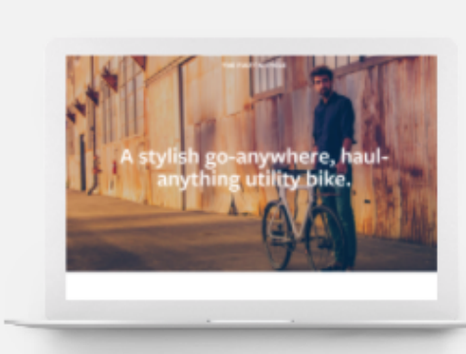
Whether you choose to make the trek down the programming path or decide to stay put, you'll have to know your way around a CMS or two. Yes, like death and taxes, they're unavoidable. So let's get you familiarized with Squarespace and Wordpress. Follow the steps below and check out the resource links as needed.

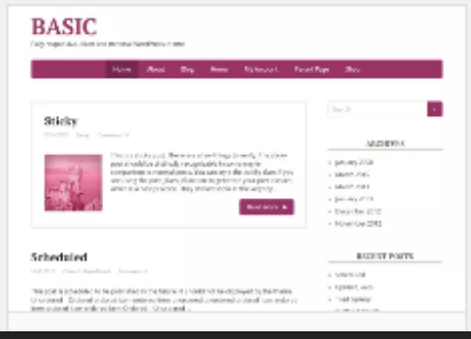
You have a client that sells international street wear. Your client communicates directly with wholesalers and doesn't keep an inventory, they drop-ship instead. If you're unfamiliar, dropshipping is when a store fulfills orders via third party (see more in resources).

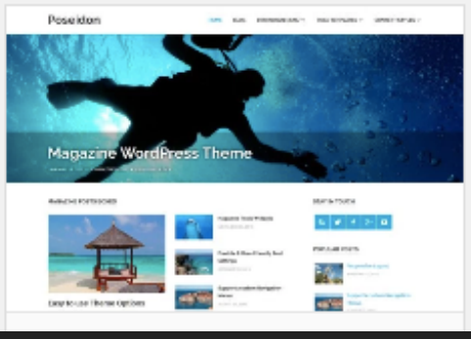
Anyway, see if you can piece together a site using Squarespace or Wordpress that would fulfill their needs. Style and design is up to you, just make sure it also functions as it should.

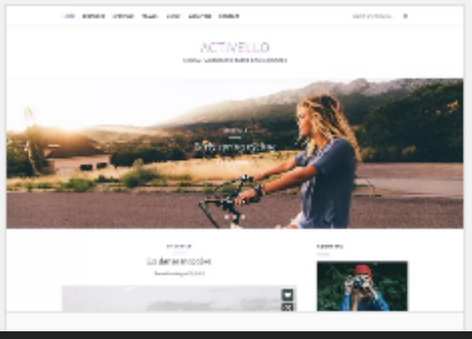












- Sign up for a Squarespace account.
- Take a look around.
- Choose a template.
- Play around with the design.

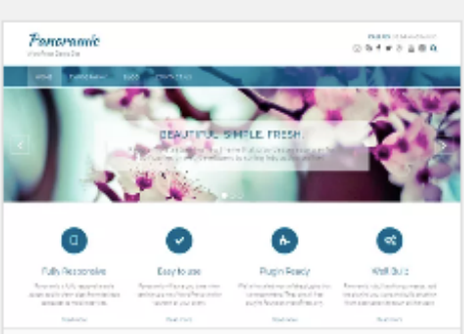
Now do the same with Wordpress.

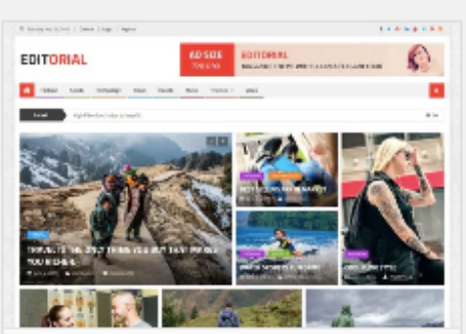
Note: There are thousands of plug-ins to choose from and you'll probably find one that says it will get the job done, but a lot of these have poorly written code or won't work with your theme.

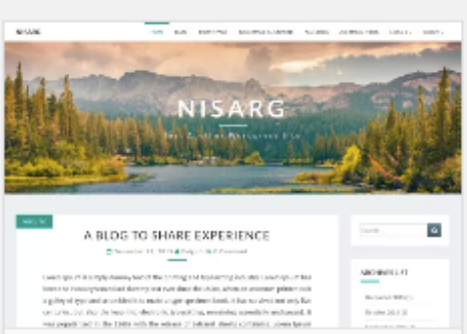
Keep the following in mind...

- Does the site do everything you want it to?
- How do you make it do all of the things?
- What are the e-commerce options?
- How do you modify a theme?
- Where is the code?
- What do you have access to?
- Can you recreate what you did in Squarespace and vice versa?

Post your site links, screenshots, or questions to the discussion board.







GITs and GIThub

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What GITs are and why they're useful

- **GIT = Online repository for saving versions of code**
- The old solution was to save them on local hard drives, but it was inefficient
- This system is easy to set up and access in the cloud
- GIT helps you keep track of what other team members are working on and what's the latest version of the code
- **GIThub = A company that took the GIT process and made their own large service for saving code**
- "Forking" = modifying a version of code to make your own version
- A lot of people fork using GITs

See you next lecture!

OOP

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What OOP is and what it does

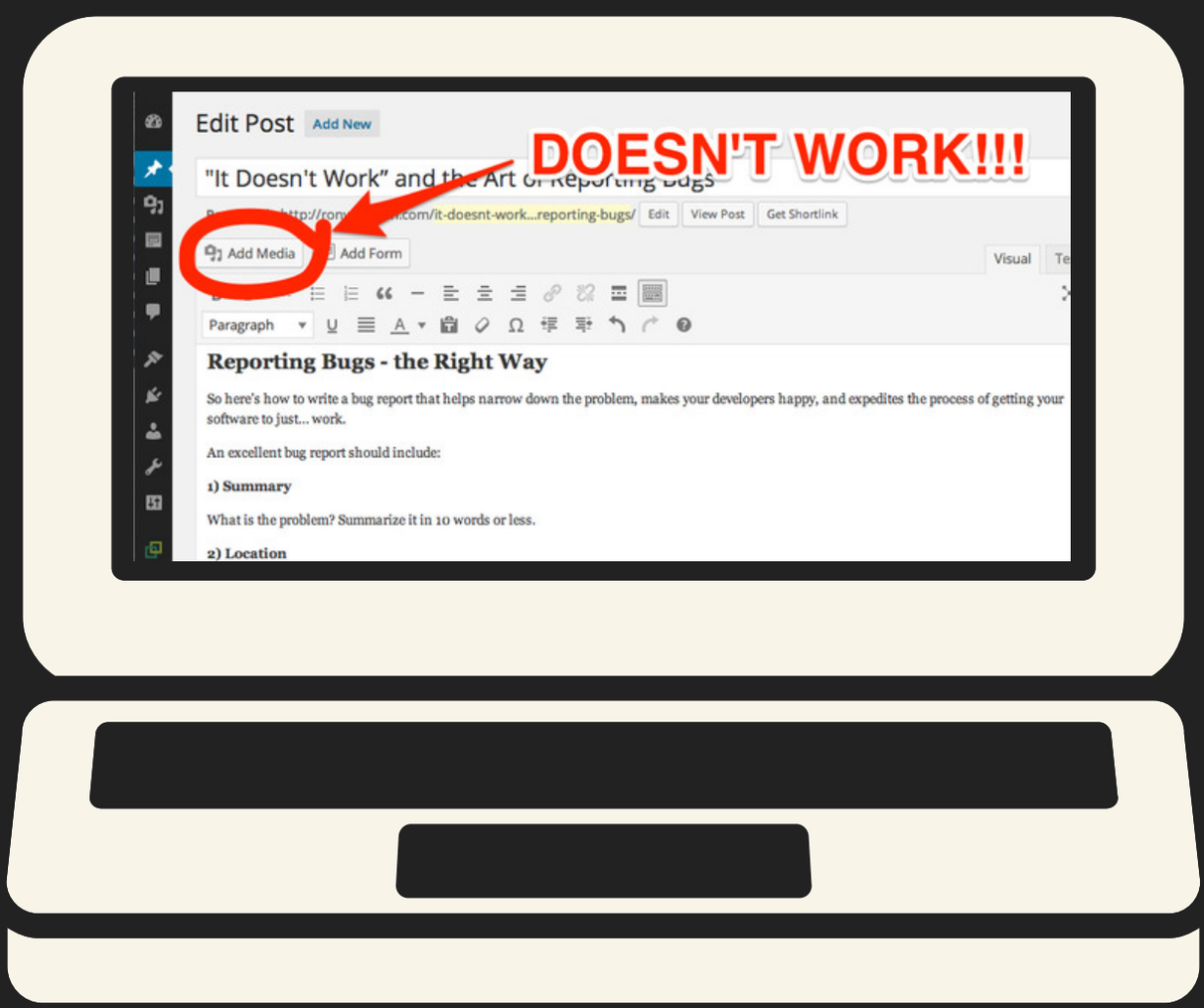
- **OOP = Object Oriented Programming**
- The standard way of developing something is called a procedural paradigm
- OOP is a new paradigm that allows you to think of programming like real objects instead of instructions
- OOP is a collection of functions and data that can be grouped into what's called an object
- When you need to access an object, you instantiate a class that contains it
- OOP helps you save time by grouping bunches of data and functions together
- You can then apply attributes broadly across one or several objects
- OOP is more scalable than procedural

See you next lecture!

A lot of people spend hours trying to find and fix every single bug when in all honesty, it's a frustrating, thankless, and never-ending endeavor - along with this activity...maybe...

Find a website that has bugs. Add a description or photo of the bug(s) to the discussion board. If you're stuck, try large e-commerce sites that have lost track of how much stuff or how many pages they have like eBay or Amazon.

"Minutes ago I was reviewing a product I'd recently bought. The review page had a little icon saying "Upload photo". So, for the first time ever, I tried to upload a photo (that compared the rather deficient product that I bought with a much better version.) Well... after 20 minutes of frustration and system freezes and hangs and liberal use of Task Manager to undo freezes... gave up on trying to upload the photo."


Edit Post
Add New
DOESN'T WORK!!!
Edit
View Post
Get Shortlink
Add Media
Add Form
Visual
Paragraph
Reporting Bugs - the Right Way
So here's how to write a bug report that helps narrow down the problem, makes your developers happy, and expedites the process of getting your software to just... work.
An excellent bug report should include:
1) Summary
What is the problem? Summarize it in 10 words or less.
2) Location

Continuous integration

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What continuous integration means

- Integration means combining two parts of code together
- Continuous integration is part of a system protocol that includes:
  - Continuous delivery
  - Continuous deployment
  - Continuous integration
- **Continuous integration is making an elaborate server setup that can automatically do the integrations for you**
- It makes everyone check in more often to see if what they're making fits with what others are making, of if there are other issues
- This system makes solving problems easier and makes the development process faster
- It's usually used by larger teams

See you next lecture!

Full stack JS

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What full stack JavaScript is

- Node.js is a system that allows you create an environment on your browser to test your JavaScript
- **JavaScript is now used as a front-end and back-end language**
- Using it for both could save time and eliminate the effort of learning other back-end languages
- Full stack JS has a lot of potential but it's still under development
- It might be inefficient and might actually take longer to write everything in it

See you next lecture!

Pair programming

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How pair programming works

- Instead of having one person program alone, you have two people standing one behind the other
- Two people will write one piece of code
- When two people work on it, you have less problems
- The process is slower
- Advantages:
  - The code quality improves
  - Developers learn from each other
- It's used mostly with big applications that have a long timeline
- It's better to do it slower but have a higher quality

See you next lecture!

Full stack design

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What a full stack designer does

- **Full stack designer = Graphic designer who can also program**
- There is a trend towards designers learning more about development
- The final product will have a better quality because they also understand what's happening behind the scenes, not just on the surface

See you next lecture!

Hybrid apps

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Pros and cons of hybrid apps

- There are three types of mobile apps:

- Native app
  - written in the code of a specific operating system
- Web app
  - a website that is redesigned to work on a smaller screen
- Hybrid app
  - designed to work on all the operating systems

- Advantage:
  Hybrid apps help save time and money

- Downsides:
  - Loss of interactivity
  - Loss of core functions and harder to control

- Native apps: Facebook, Twitter
- Hybrid apps: Netflix

See you next lecture!

Responsive design

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What it is and why we use it

- Nowadays, phones and computers come in a lot of different sizes
- You have to make sited that looks good on all resolutions
- **Responsive design is a design that detects the size of the screen and adapts all the elements on the app to that resolution**
- You have to decide what gets displayed on lower resolutions
- Usually, responsive design has a lot of white space on the sides that allows it to contract without losing elements
- The elements of the site have a relative position (floating)
- You need to run a lot of bug testing on different resolutions and browsers

See you next lecture!

SaaS, PaaS, & IaaS

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What they are and how they work

- "In the cloud" = Stored and accessed on a different device or server than the device you're using
- People use cloud storage in order to have access to their files at all times

▶ **#1 SaaS = Software as a Service**

Products become services by providing a web version

You pay per month and you login to use the online app

Advantages:

- You charge less, but more often, and for longer
- People can access files everywhere
- Continuous updating

▶ **#2 PaaS = Platform as a Service**

A platform is a set of tools, resources, and software that come together to let you build software on top of it

- Example: creating games on Facebook

- **#3 IaaS = Infrastructure as a Service**

  IaaS is what holds up and contains whatever you're building

  Web storage is an IaaS

  Iaas systems offer:

  - Security and firewalling

  - Routing and network services

  - Maintenance and load balancing

- Saas runs on Paas, which runs on an Iaas - like a stack

See you next lecture!

SWIFT

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

What Swift is

- **Swift is a new programming language designed by Apple specifically for their operating systems**
- In the past, you would build iOS apps in Objective C, which is harder to learn and more strict
- Currently you can make an app in either of them
- Swift is safer to use, fun, and faster
- Swift is not just for iOS, it also works for Apple Watch, TVOS, Mac OS
- Apple wanted to accelerate the developer community for their system by creating Swift
- Swift is not that easy to learn

See you next lecture!

Web developer vs. software engineer vs. full stack developer vs. hacker

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

The difference between the four concepts

- **#1 Web developer**

  - Someone who has a strong competency in front-end design, with basic competency in back-end programming
  - They can make a blog or a website
  - They focus on how it looks
  - They can be specialized in one of the platforms, like WordPress

- **#2 Software engineer**

  - Someone who has a strong competency in back-end development and basic competency in front-end development
  - They focus on complex apps
  - They solve really hard problems and are paid a lot of money

- **#3 Full stack developer**

  - Someone who is good at both front-end and back-end development
  - They are very rare
  - Usually, they are just ok at both

- **#4 Hacker**

  - Someone who has basic competency in both front-end and back-end programming
  - They can hack together something pretty quickly
  - They try to get something done and they are not focused on usability

See you next lecture!

How do you choose?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Questions to ask yourself

- **#1 What are you orientated towards?**

  - Front-end developers will work a lot with design and will have to be creative
  - Back-end developers will solve problems and use logic

- **#2 Where do you see yourself using this programming?**

  - Where do you want to work?
  - Big companies need back-end developers
  - If you want to work on your own projects, focus on the things that get it done the fastest (ex. Python, Ruby)
  - You can hire someone to do the front-end
  - If you like mobile, learn how to do mobile apps

See you next lecture!

How to learn front-end

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Where to start

- Start with HTML because it's the easiest
- w3schools.com, Youtube tutorials
- Then move on to CSS
- Use online courses
- Download websites and try to edit and change the CSS
- Get an introduction to JavaScript
- Codecademy, Tutsplus, Lynda, Treehouse, OneMonth, Bento
- Learn JQuery
- Start looking at front-end development bootcamps (mostly online)

See you next lecture!

How to learn back-end

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

Where to start

- Back-end has the most options for learning
- First, pick a language
- Look at the companies you'd like to work for and see what language they use
- Ruby is best for web development and startups
- Python is the most flexible, you can program on desktop and the web
- PHP is best if you want to start your own thing or build something quickly
- Java is good if you want to work in a large company or want to build Android apps

See you next lecture!

PHP, Ruby, Python

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

How to start learning each language

- **#1 Ruby**

  - Get familiar with the syntax and how it works
  - Start with Codecademy
  - Check out the Ruby community
  - tryruby.org, rubymonk.com, codewars.com
  - Online courses: OneMonth, Treehouse
  - Get familiar with Rails

- **#2 Python**

  - Google has an online course for it
  - Codecademy, Coursera
  - Learn a framework (examples: Django, Scrapy)

#3 PHP

- Make sure you learn the most recent version

- You have to know basic HTML and CSS

- Codecademy, Udemy, books

- Understand how to use PHP in an object oriented way (OOP)

See you next lecture!

Should I specialize in new technologies?

LECTURE SUMMARY

PRE-PROGRAMMING

Covered in this lecture:

More advice on how to choose

- The technology is constantly changing
- It doesn't mean that all the current trends will stick on the long term
- **Stick with the stuff that has persisted over time**
- What technology you use has no value to the clients
- If you have a personal preference, choose that one
- Keep up to date with changes in the industry

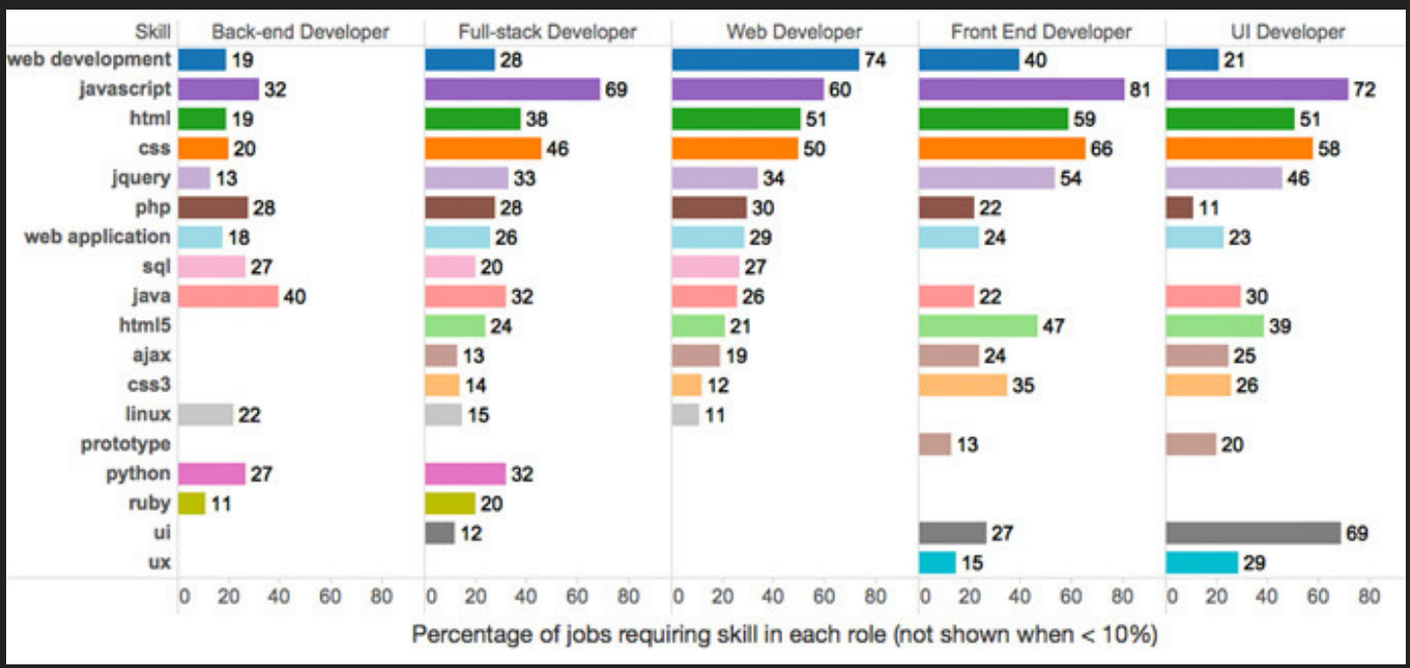
See you next lecture!

If you're taking this course then you're a long ways off, but it doesn't hurt to check out the job boards to see what you should or shouldn't be focusing on. Authentic Jobs, Dribbble, Angel List, GitHub, and StackOverflow are good places to begin.

But first thing's first, do you know what the following job titles mean? The lines are blurred because the industry is rapidly evolving -- meaning new skills, new roles, and are frontend and backend supposed to be two words? Is there a dash? What's the deal?

- *Frontend developer (aka Front End Web Developer, HTML Developer)*
- *Backend Developer (aka Backend Engineer, Backend Software Developer)*
- *Interaction designer*
- *Software engineer*
- *Full-stack Developer (aka Full Stack Engineer, Full Stack Software Engineer)*
- *Backend generalist engineer*
- *Program developer*
- *Web Developer (aka Web Applications Developer, .NET Web Developer, Web Engineer, PHP Web Developer, Web Programmer, Web Architect)*
- *Frameworks specialist*
- *UI Developer (aka User Interface Developer, UI/UX Developer, Web User Interface Developer, User Interface Design Engineer)*
- *Wordress developer*
- *UX Designer*

So what skills do you need to know? What are the essentials? This chart (2014) might clarify a bit.


Skill
Back-end Developer
Full-stack Developer
Web Developer
Front End Developer
UI Developer
web development 19 28 74 40 21
javascript 32 69 60 81 72
html 19 38 51 59 51
css 20 46 50 66 58
jquery 13 33 34 54 46
php 28 28 30 22 11
web application 18 26 29 24 23
sql 27 20 27
java 40 32 26 22 30
html5 24 21 47 39
ajax 13 19 24 25
css3 14 12 35 26
linux 22 15 11
prototype 13 20
python 27 32
ruby 11 20
ui 12 27 69
ux 15 29
0 20 40 60 80
Percentage of jobs requiring skill in each role (not shown when < 10%)

One last thing...

Do some digging around on the aforementioned job boards. Do any catch your eye? Any particular skills or languages? Or are you still confused? (It's ok if you are, this is an introductory course, after all.)

Choose your potential dream job and tell us why. Then add a few notes about how you might land it.

Take your job links, questions, answers, and confusions to the discussion board.